Planning and Scheduling (SWEN90016)

Shanika Karunasekera
Department of Computing and Information Systems
University of Melbourne
karus@unimelb.edu.au

Overview

- **Project Plan**
- **Basic Planning Concepts**
- **People and Effort**
- **Project Scheduling**
  - Work breakdown
  - Dependencies
  - Task Networks
  - PERT and Gantt
  - Critical Path Methods

Project Plan

- **What does it include:**
  - the tasks that need to be carried out as part of the processes that are being followed
  - the duration and dependencies for each task
  - the people and physical resources required by each task
  - milestones or goals of each task

Basic concepts related to planning

- **Compartmentalise**
- **Interdependency**
- **Effort Estimation/Validation**
- **Time Allocation**
- **Responsibilities**
- **Outcomes/Goals**
- **Milestones**

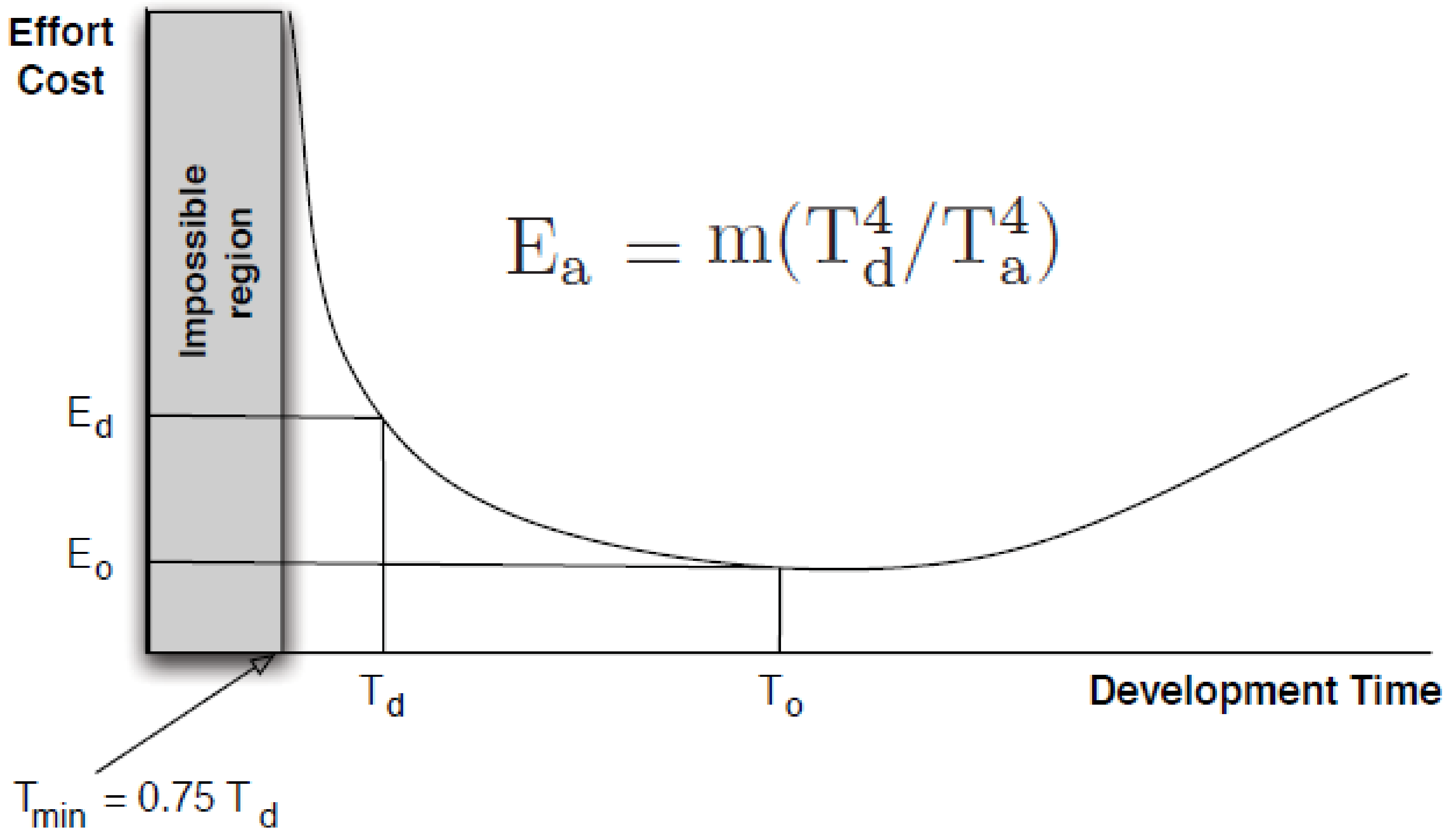
People and Effort

- **A common measure for estimating the effort for software is *man-months* (more generally *person-months*)**

- **person-months:**
  - the time in months for a single person working full time to complete the task

- **The Mythical Man-Months [Brooks seminal paper]**
  - man-months is a misleading measure to estimate software
  - adding people to a project that is behind schedule could result in more damage than helping it

People Effort



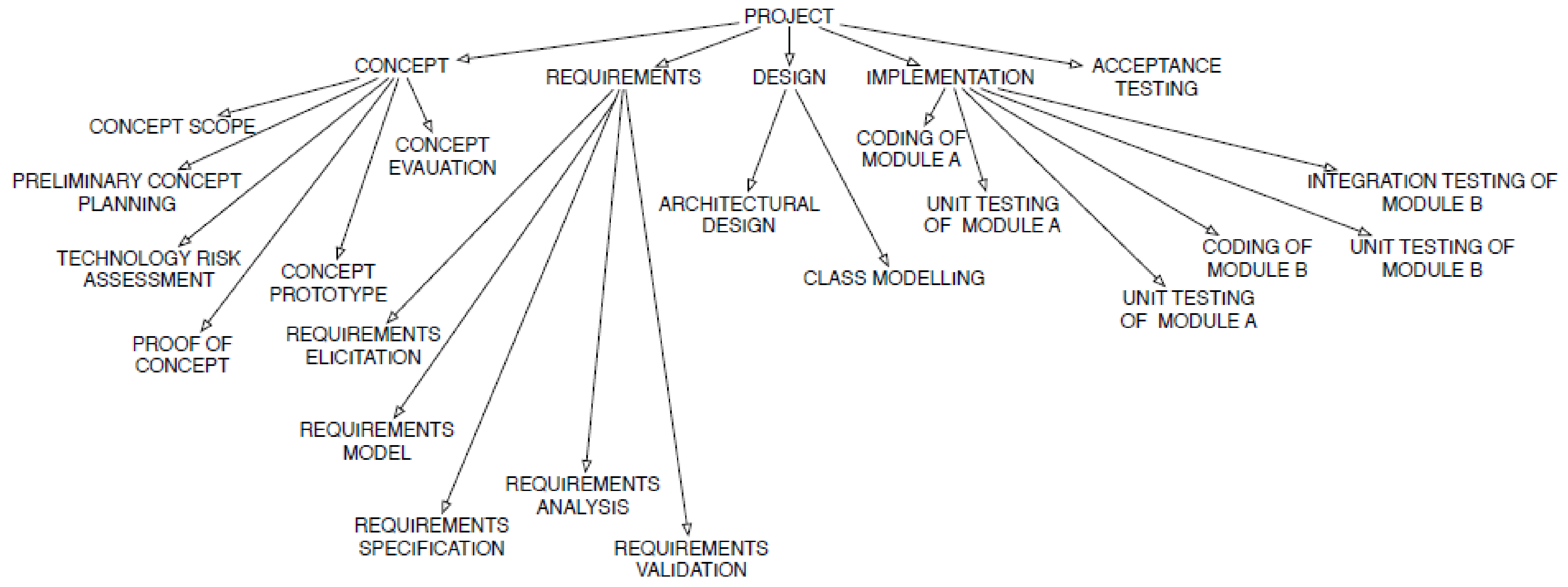
$$E_a = m(T_d^4 / T_a^4)$$

Putnam-Norden-Rayleigh curve

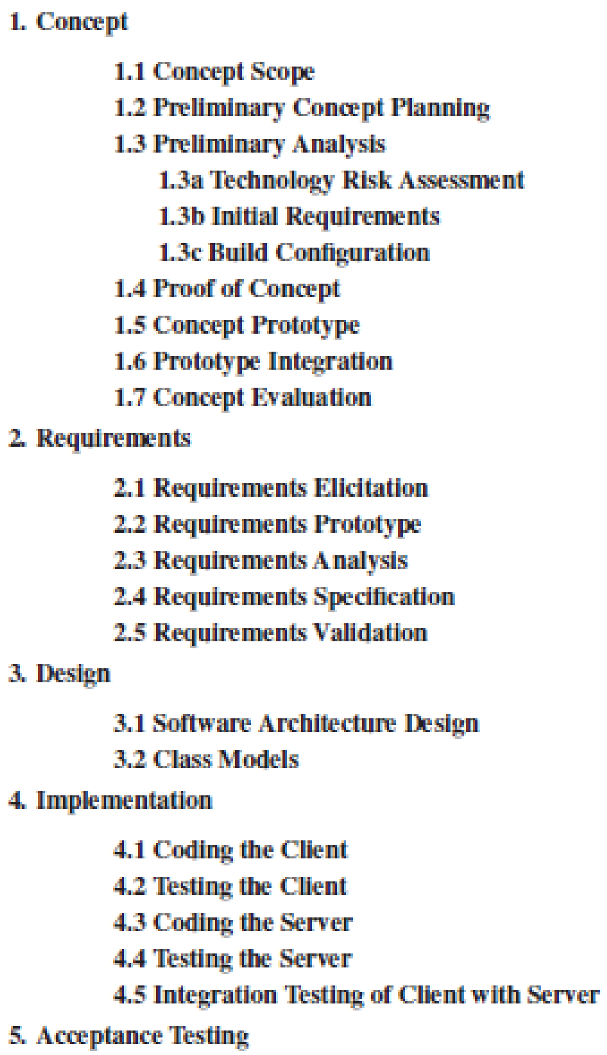
Work Breakdown

- **Start by choosing the SDLC**
- **Breakdown the work tasks - *Work Breakdown Structure***
  - e.g. waterfall model
    - Concept
    - Requirements
    - Design
    - Implementation
    - Acceptance Testing
- **100% rule**
  - Work breakdown structure includes100% of the work defined by the project scope and captures all deliverables — internal, external, and interim — in terms of the work to be completed, including all project management.

Work breakdown structure



Work breakdown structure



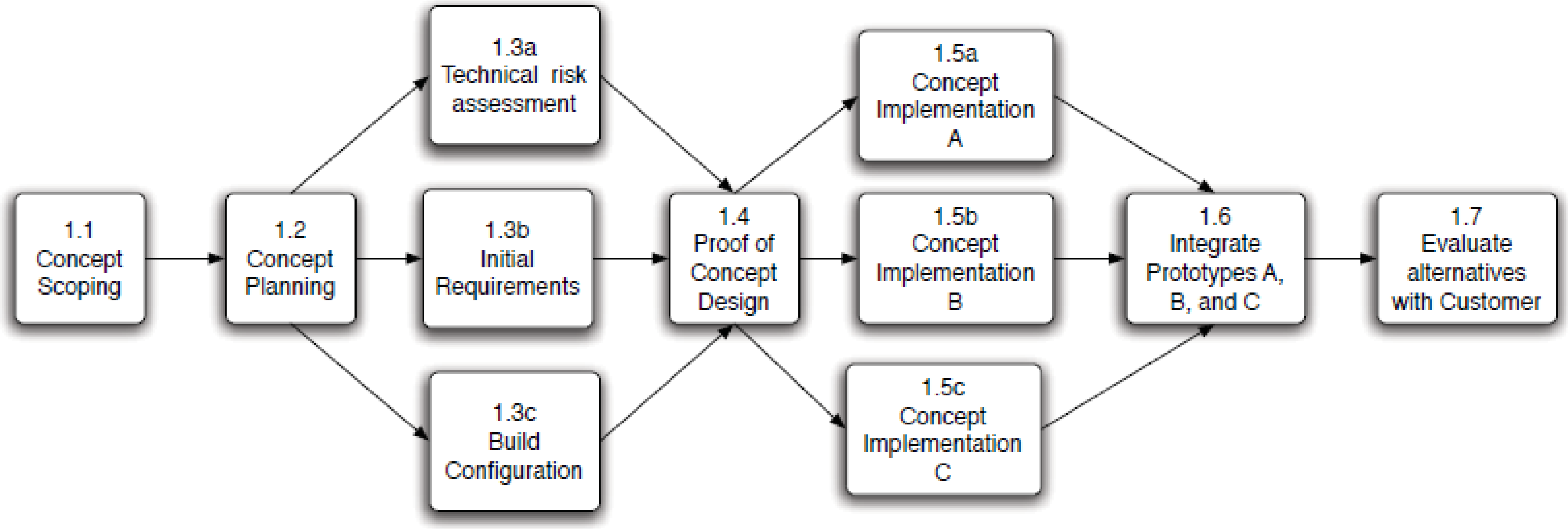
1. Concept
   - 1.1 Concept Scope
   - 1.2 Preliminary Concept Planning
   - 1.3 Preliminary Analysis
     - 1.3a Technology Risk Assessment
     - 1.3b Initial Requirements
     - 1.3c Build Configuration
   - 1.4 Proof of Concept
   - 1.5 Concept Prototype
   - 1.6 Prototype Integration
   - 1.7 Concept Evaluation
2. Requirements
   - 2.1 Requirements Elicitation
   - 2.2 Requirements Prototype
   - 2.3 Requirements Analysis
   - 2.4 Requirements Specification
   - 2.5 Requirements Validation
3. Design
   - 3.1 Software Architecture Design
   - 3.2 Class Models
4. Implementation
   - 4.1 Coding the Client
   - 4.2 Testing the Client
   - 4.3 Coding the Server
   - 4.4 Testing the Server
   - 4.5 Integration Testing of Client with Server
5. Acceptance Testing

Dependencies

- **Dependencies are caused by:**
  - a task needing a work product of another task
  - a task needs resources used by another task

- ***Task Network*** **captures the dependencies between tasks**

Task Network


1.1 Concept Scoping
1.2 Concept Planning
1.3a Technical risk assessment
1.3b Initial Requirements
1.3c Build Configuration
1.4 Proof of Concept Design
1.5a Concept Implementation A
1.5b Concept Implementation B
1.5c Concept Implementation C
1.6 Integrate Prototypes A, B, and C
1.7 Evaluate alternatives with Customer

Project Schedule

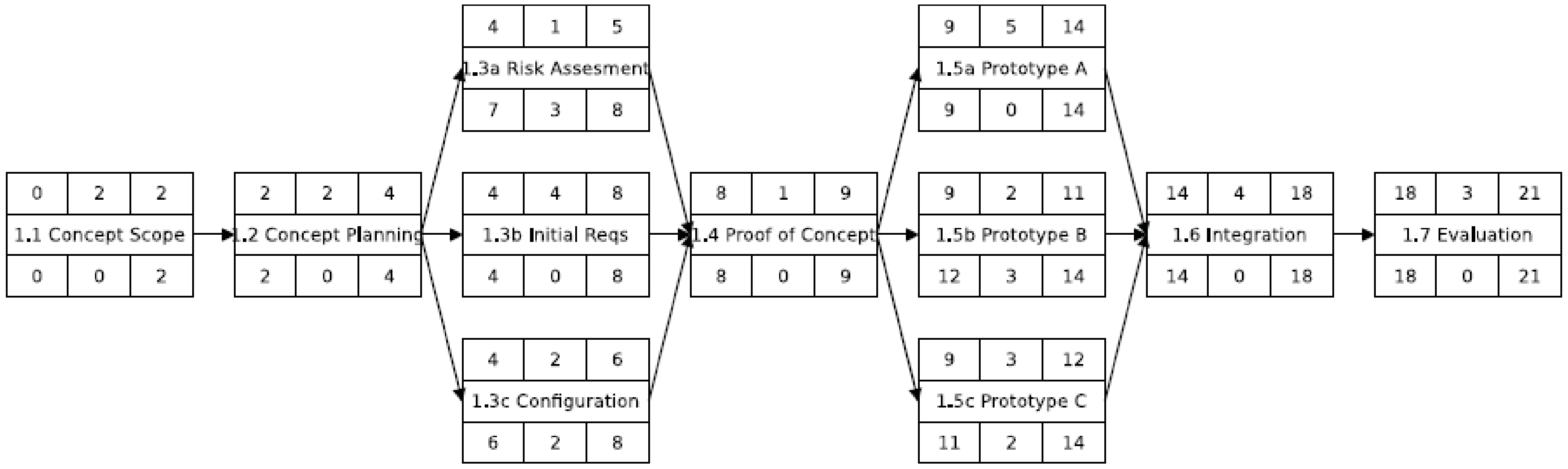
- **Two important questions not answered so far:**
  - How long will the system take to develop?
  - How much will it cost?

- **Two widely used graphical notations**
  - PERT (Program Evaluation and Review Technique) charts
    - An activity network that shows the dependencies among tasks and the *critical path*.
  - Gantt charts
    - A bar chart that show the schedule against a calendar

Important concepts

- Milestone
- Activity
- Free float, free slack
- Total float, total slack
- Critical path
- Critical activity

PERT Charts



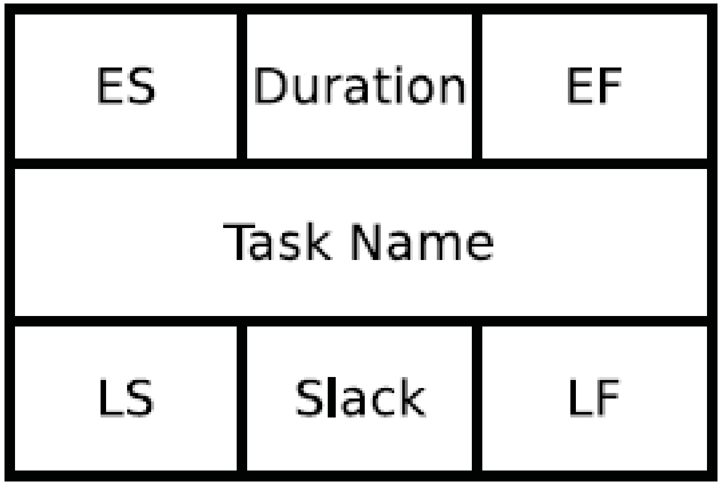
0 | 2 | 2
1.1 Concept Scope
0 | 0 | 2
2 | 2 | 4
1.2 Concept Planning
2 | 0 | 4
4 | 1 | 5
1.3a Risk Assesment
7 | 3 | 8
4 | 4 | 8
1.3b Initial Reqs
4 | 0 | 8
4 | 2 | 6
1.3c Configuration
6 | 2 | 8
8 | 1 | 9
1.4 Proof of Concept
8 | 0 | 9
9 | 5 | 14
1.5a Prototype A
9 | 0 | 14
9 | 2 | 11
1.5b Prototype B
12 | 3 | 14
9 | 3 | 12
1.5c Prototype C
11 | 2 | 14
14 | 4 | 18
1.6 Integration
14 | 0 | 18
18 | 3 | 21
1.7 Evaluation
18 | 0 | 21

PERT Charts

- **Terminology**
  - predecessor node
  - successor node
  - optimistic time (O)
  - pessimistic time (P)
  - most likely time (M)
  - expected time (TE)

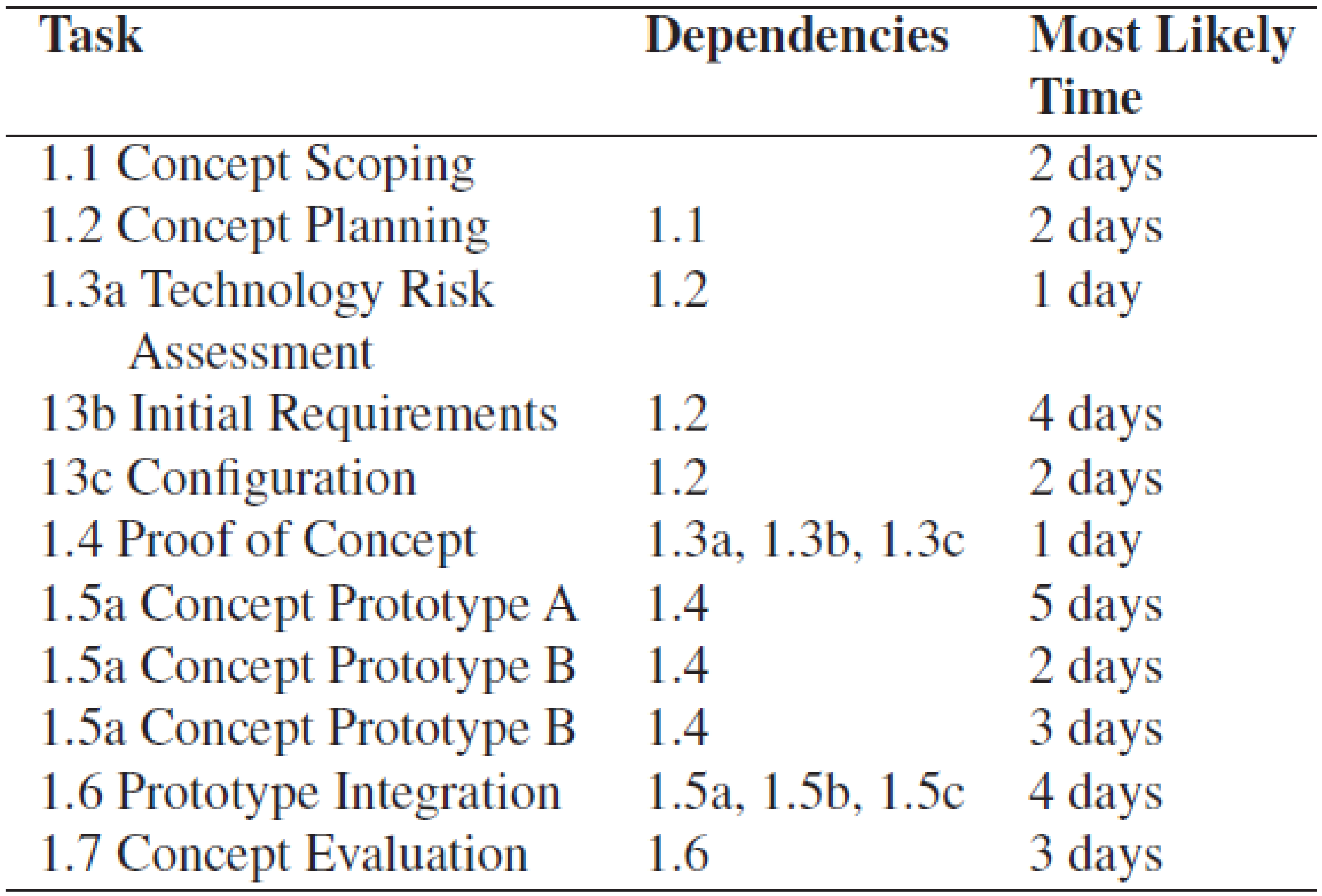
$$\mathrm{TE} = (\mathrm{O} + 4\mathrm{M} + \mathrm{P})/6$$

PERT Charts - node



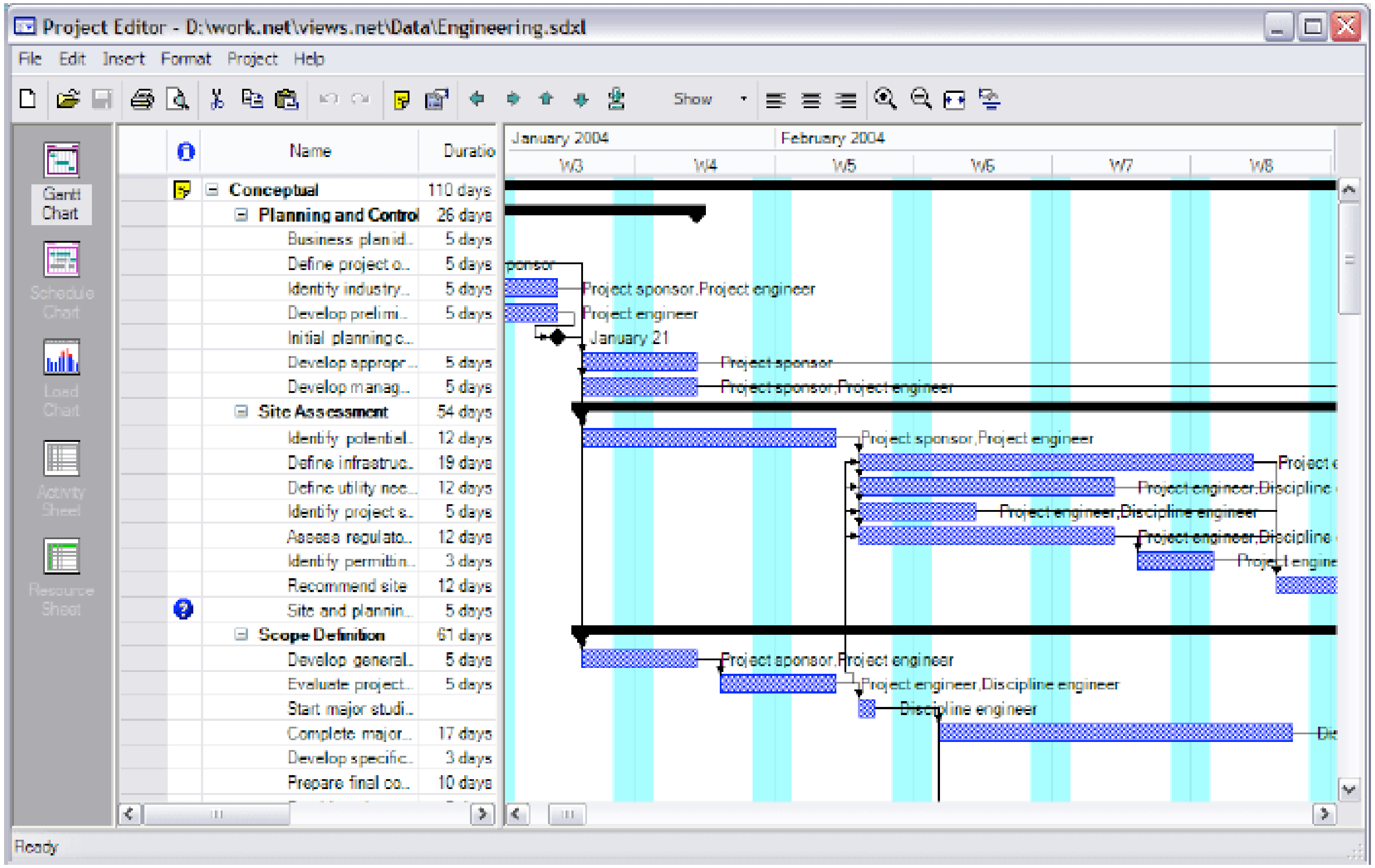
ES	Duration	EF
Task Name		
LS	Slack	LF

PERT Charts - dependencies



Task	Dependencies	Most Likely Time
1.1 Concept Scoping		2 days
1.2 Concept Planning	1.1	2 days
1.3a Technology Risk Assessment	1.2	1 day
13b Initial Requirements	1.2	4 days
13c Configuration	1.2	2 days
1.4 Proof of Concept	1.3a, 1.3b, 1.3c	1 day
1.5a Concept Prototype A	1.4	5 days
1.5a Concept Prototype B	1.4	2 days
1.5a Concept Prototype B	1.4	3 days
1.6 Prototype Integration	1.5a, 1.5b, 1.5c	4 days
1.7 Concept Evaluation	1.6	3 days

Gantt Chart


Project Editor - D:\work.net\views.net\Data\Engineering.sdxl
File Edit Insert Format Project Help
Show
Gantt Chart
Schedule Chart
Load Chart
Activity Sheet
Resource Sheet
January 2004
February 2004
W3
W4
W5
W6
W7
W8
Project sponsor,Project engineer
Project engineer
January 21
Project sponsor
Project sponsor,Project engineer
Project engineer,Discipline engineer
Discipline engineer
Ready

	Name	Duratio
	⊟ **Conceptual**	110 days
	⊟ **Planning and Control**	26 days
	Business planid..	5 days
	Define project o..	5 days
	Identify industry..	5 days
	Develop prelimi..	5 days
	Initial planning c..	
	Develop appropr...	5 days
	Develop manag...	5 days
	⊟ **Site Assessment**	54 days
	Identify potential..	12 days
	Define infrastruc..	19 days
	Define utility nee...	12 days
	Identify project s..	5 days
	Assess regulato..	12 days
	Identify permittin..	3 days
	Recommend site	12 days
	Site and plannin..	5 days
	⊟ **Scope Definition**	61 days
	Develop general..	5 days
	Evaluate project..	5 days
	Start major studi..	
	Complete major..	17 days
	Develop specific..	3 days
	Prepare final co..	10 days

Critical Path Methods

- **Critical Path**
  - path with the longest duration
  - activities on the critical path have a total free slack of 0
  - a delay in any of the activities in the critical path will cause the project to delay

- **Crashing the project plan:**
  - shortening the total duration of the project by shortening the critical path
    - By removing the dependencies between activities in the critical path; or
    - Shortening the duration of activities in the critical path

Project Tracking and Control

- Period reviews where team members report progress
- Evaluating the results of reviews and audits conducted as part of the software engineering process
- Tracking formal project milestones
- Comparing actual start dates with scheduled start dates
- Meeting engineers and having informal discussions
- Using a formal method like ***earned value analysis***

Common reasons for project failure

- **Unrealistic deadlines**
- **Changing requirements**
- **Underestimate of the efforts**
- **Unmanaged risks**
- **Technical difficulties**
- **Human resource difficulties**
- **Failure to see and act on slippage**
- **Miscommunications between project staff**

Planning in agile development

- **Takes a significantly different flavour from traditional approaches**
- **Detailed planning is differed until the start of the iteration**
  - Designed to handle change
  - An iteration includes all phases (requirements, design and test)

- **Planning is based on light weight lists**
  - Gantt and PERT charts are considered less useful

Planning in agile development

- Plan short iterations
- Produce useful functionality
- Use “Just in time (JIT) planning” – next iteration
- Use the team